September 1983

Low-Cost CRT Control Does More with Less

Thomas Rossi, Applications Manager
Peripheral Components Division

Reprinted with permission from Electronic Design', Vol.29, No.9; copyright Hayden Publishing Co., Inc., 1981

ORDER NUMBER: 210507-001

Fewer parts make a microprocessor-based CRT controller cost-effective, and interrupt-driven software cuts overhead on the system's CPU.

Low-cost CRT control does more with less

The multitude of components and the CPU overhead long associated with cathode-ray-tube controllers are rapidly becoming conspicuous by their absence. In particular, an intelligent terminal based on Intel's iAPX 88/10 (8088) microprocessor and 8276 small-system CRT controller eliminates all but 22 of the nearly 40 chips required by other CRT controllers (even those with microprocessors and integrated peripherals). It also cuts overhead on the processor to less than 25%, so that the 88/10 is free to implement such intelligent terminal functions as local data processing.

The iAPX 88/10 implementation supplies characters directly to the 8276 by means of interrupt-driven software, eliminating the need for a direct-memory-access (DMA) controller. The design interfaces directly with standard CRT monitors, contact-closure keyboards, and RS-232C serial-communication links (asynchronous or bisynchronous), to provide a complete stand-alone operator interface.

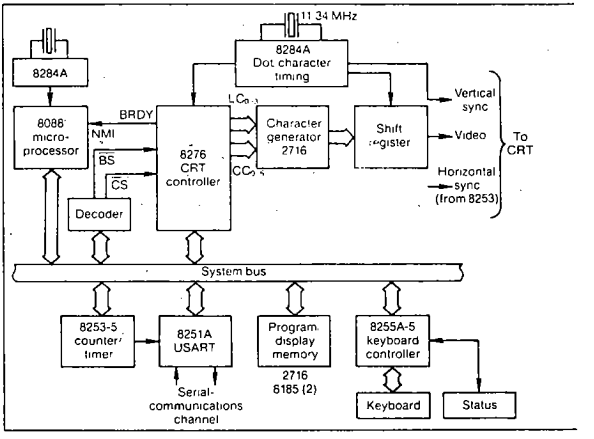
Although the primary design goal—implementing a low-cost CRT terminal—has excluded some useful CRT features, these are easily made available through additional external hardware. For example, composite video is added with two TTL packages, a transistor, and some resistors and capacitors. Another simple option involves the two general-purpose attribute outputs on the 8276 and lets users select any one of four colors on a color monitor.

Basic system configuration and architecture

Central to the 22-chip CRT controller design is an iAPX 88/10 8-bit microprocessor operating at 5 MHz and supported by two 8185 1-kbit × 8 static RAMs and a 2716 control software PROM (Fig. 1). An 8251A programmable communication interface provides synchronous or asynchronous serial communications. Three manual switches on the PC board select the baud rate, and one of the 8253's three independent programmable interval timers generates the 8251A's baud-rate clock under software control.

The three PC-board switches are monitored by the iAPX 88/10 to determine the desired baud rate. When the CPU detects a change in the switch positions, the 8253 is loaded with the appropriate count for the new baud rate.

An 8255A provides three 8-bit parallel I/O ports. Two I/O ports contribute keyboard scanning, and the



1. Intelligent terminals, built with Intel's iAPX 88/10 (8088) microprocessor and new 8276 small-system CRT controller, take this basic configuration to reduce parts count and minimize overhead on the system CPU.

Thomas Rossi, Applications Mgr. Peripheral Components / Intel Corp.
3065 Bowers Ave., Santa Clara, CA 95051

third port senses option-switch settings and the vertical-retrace signal from the 8276 (for CRT synchronization upon reset).

The CRT dot and character timing is generated by an 8284A clock generator. Another 8253 timer provides the appropriate horizontal-retrace timing for the CRT monitor. In its programmable one-shot mode, this timer generates a 32-μs horizontal-retrace pulse for the CRT monitor (Ball Brothers TV-12). A simple user-initiated change in the software will modify this delay time to suit different CRT monitors. The third and last timer in the 8253 is available for any user-defined need.

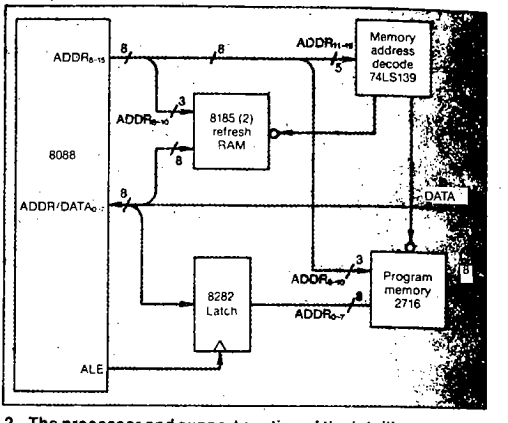
A 2716 EPROM on the controller board serves as a user-programmable character generator. A shift register transforms the data from the character EPROM into a serial-bit stream to illuminate dots on the CRT screen. The 2716 character generator helps to create special symbols and characters for word processing, industrial-control applications, or foreign-language displays.

The controller hardware is divided into processor and support, serial and parallel I/O, and CRT-control sections. The processor and support section consists of an iAPX 88/10 microprocessor, which is supported by two 8185 1-kbit × 8 static-RAM devices, and another 2716 EPROM (containing 2 kbytes of control firmware). The iAPX 88/10 uses a 15-MHz crystal (with an 8284A) to operate at a 5-MHz clock rate. The 8185 memories attach directly to the iAPX 88/10 multiplexed bus. An 8282 latches eight address lines (A_0-A_7) from the multiplexed bus for 2716-program memory access (Fig. 2).

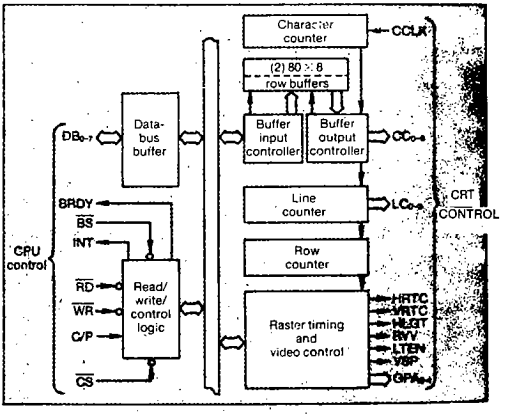
The serial and parallel I/O section of the terminal includes the 8255A programmable peripheral interface, and the CRT section contains the 8276 CRT controller and support circuits. All of the controller's I/O operations are memory mapped (see table).

Memory map of controller I/O operations

Address range	Selected device	Comments
00000 - 00003	RAM	Interrupt vector
00004 - 00029	RAM	Stack, local variables
00030 - 007FF	RAM	Display buffer
01000 - 01001	8276 $\overline{CS}$	8276 command/status
01900	8276 $\overline{BS}$	8276 row buffers
12000 - 12001	8251A	Serial channel
14000 - 14003	8253	Baud-rate timer
18000 - 18003	8255A	Keyboard, switches
FF800 - FFFFF	2716	Program storage



2. The processor and support section of the intelligent terminal's hardware contains two 8185 RAMs attached directly to the iAPX 88/10 multiplexed bus. An 8282 latches eight address lines (A_0-A_7) from the multiplexed bus for 2716-program memory access.



3. Here are the major functional blocks of the 8276 programmable CRT controller. This device permits software specification of most CRT-screen format characteristics (cursor position, characters/row, rows/frame).

How the controller board communicates

The CRT-controller board communicates to computer systems and other CRT units through a serial interface. Both RS-232C and TTL-compatible interfaces are available at the J_1 connector. The unit's standard software supports eight data-transmission rates: 9600, 4800, 2400, 1200, 600, 300, 150, and 110 baud. These rates are switch-selectable on the PC board. Since the baud-rate clock is generated by an 8253, baud rates may be easily modified in software.

Keyboard scanning is supported through the A and B ports of a 8255A programmable peripheral interface. Therefore, low-cost unencoded keyboards can be used. The eight scan lines (port B) and eight return lines (port A) support a 64-contact closure-key matrix. The three switches attached to port C permit baud-rate selection. Four general-purpose

inputs on port C permit the software to sense depression of the caps-lock key, the control key, and the shift key, as well as the position of the line/local switch. The last input on port C senses the status of the vertical retrace (VRTC) output of the 8276, so that the controller can synchronize with the CRT display on power up or after a hardware reset.

All keyboard I/O connects to the terminal board by means of a 40-pin header on its edge. All seven option-switch inputs are also brought to the connector, so that option switches may be installed on the keyboard if desired.

Software specifies the screen format

The CRT display is controlled by the 8276 programmable CRT controller (Fig. 3). With this device, most CRT screen-format characteristics—such as the cursor position, the number of characters per row, and the number of rows per frame—can be specified through software. The 8276 handles all display timing including retrace time delays.

In the current design, 2000 characters are displayed on the CRT screen (25 rows of 80 characters). Each character is formed as a 5×7-dot matrix within a larger 7×10 matrix (Fig. 4). Other screen formats (e.g., 16 rows of 64 characters) can be easily implemented with a few software changes and no hardware changes.

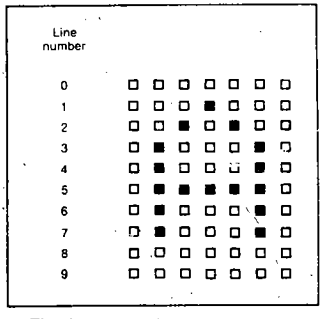
The 8276 contains two 80-character row buffers (see "Row Buffers Reduce System Overhead"). While one buffer displays the current character line on the screen, the 8276 fills the other row buffer from memory. This data transfer begins when the 8276 issues a data request (by means of the BRDY pin), causing an interrupt to the CPU. In response to this interrupt, the CPU activates the RAM's CS and RD inputs, while simultaneously activating the 8276 BS and WR inputs (Fig. 5). Through this technique, a single bus cycle suffices to transfer each byte from the RAM into the CRT row buffer. After the row buffer is filled, the CPU exits the interrupt-service routine.

But the 8276 can do more than simply paint characters on a CRT screen. Its end-of-row-stop buffer-loading code allows the control software to blank individual display lines. Also, the end-of-the-screen-stop buffer-loading code initiates an erase to the end of the screen.

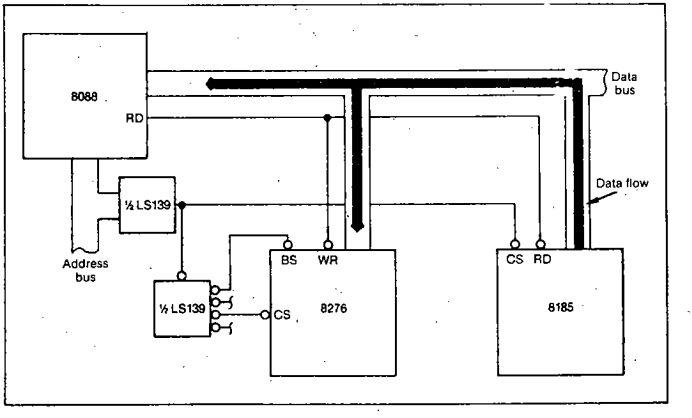
The 8276 supports software selection of visible-field "attributes" that can blink, underline, or highlight (intensify) characters on the screen and can reverse the video-character fields (black letters on a white background). Two general-purpose attribute outputs are provided to control the user-defined display capabilities.

Hardware provides three support functions

The 8276 is supported by three hardware functions: a dot/character-clock oscillator, an EPROM character generator, and a character-shift register (Fig. 6). The dot/character-clock oscillator consists of an 8284A operating at 11.34 MHz and providing an 88.2-ns dot clock. A 74LS163 divides this clock by 7 to generate a 1.62-MHz (617-ns) character clock.



4. The dot-matrix character font used in the low-cost CRT controller creates a 5×7 character in a 7×10 matrix (example shown is an upper-case A). Top and bottom lines are blanked for character separation, and the remaining line is reserved for cursor/underline display.



5. Row-buffer loading for the 8276 begins when a single 8088 string instruction moves data bytes from the 8185 RAM to the 8276 row buffer. The 8088 CPU "thinks" it is loading the AX register.

The 8276 is programmed to display one raster line every 61.7 μs—a complete character line every 617 μs (ten raster lines). The 8276 is also programmed to refresh the screen every 16.7 ms (60 Hz).

Each character row consists of ten raster lines. Seven lines display the 5 × 7-character matrix, two lines are blanked for row spacing, and one line displays the cursor and underline.

The 8276 uses the line count (LC_0-LC_3) outputs to indicate the current raster line during the display of each character. These outputs, combined with the character-code outputs (CC_0-CC_6), are sent to the 2716, which generates the dot pattern for display. This dot pattern is loaded into the shift register and is serially clocked for display by the 11.34-MHz dot clock.

During the vertical-retrace interval, the row buffer for the first line of the next frame is loaded by the iAPX 88/10. When the frame starts, the 8276 outputs the first character on its CC_0-CC_6 pins; the LC outputs are all zero. Exactly 617 ns later, the next character code is emitted by the 8276. This process continues every 617 ns until all 80 characters have been output. Then the 8276 generates a horizontal-retrace pulse, which is converted to the appropriate pulse width for the CRT monitor by the 8253.

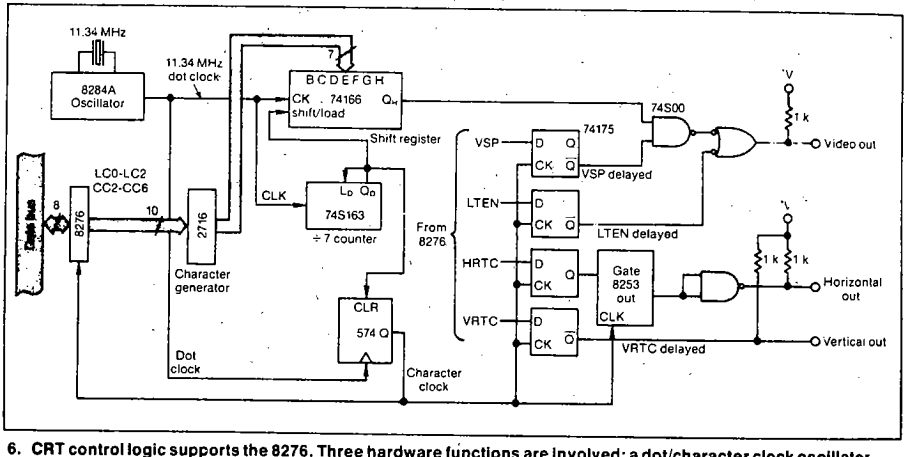
At the end of the first raster line, the 8276 increments the LC outputs. The next nine raster lines are similar to the first—the 8276 outputs the same 80 character codes on the CC_0-CC_6 pins for each of the raster lines, and the LC outputs are incremented after each raster line.

While the ten raster lines are being displayed, the 8276 is also filling the next row buffer. After the tenth raster line is completed, the 8276 resets the LC count and outputs character codes for the second row on the CC_0-CC_6 pins. As this row is displayed, the first row buffer is filled with information for the third row. The 8276 alternates row buffers until all 25 rows are displayed. At this time, the vertical-retrace signal is activated, and the scanning process is repeated for the next frame.

During display, the 8276 automatically activates the video-suppress pin (VSP) and/or light-enable outputs (LTEN), as appropriate, to control retrace blanking, generate the cursor, or underline characters.

Software is split between two priorities

The software for the CRT controller is divided into high and low-priority sections. The high-priority "foreground" software is activated each time the 8276 requests (through the iAPX 88/10 NMI interrupt) that an 80-character row buffer be filled. The 8276 row buffer is filled by performing 80 sequential memory reads. As each read is performed, the



6. CRT control logic supports the 8276. Three hardware functions are involved: a dot/character clock oscillator, an EPROM character generator, and a character-shift register.

Row buffers reduce system overhead

If no row buffer is present, the CRT controller must go to main memory to fetch every character during every dot scan line. Thus, the central processing unit is forced to relinquish the system bus 90 to 95% of the time. That CPU inactivity (overhead) greatly degrades total system performance and efficiency. CRT terminals using this approach are typically limited to between 1200 and 2400 baud on their serial-communications channels.

However, with the 8276's row-buffered architecture, the CRT controller need only access the main memory once for each displayed character row. This approach reduces system bus overhead for CRT refreshing to 25% (maximum). The CPU is then free to perform other local-processing functions, for instance, processing data at 9600 baud on a serial-communications channel.

```
        PUSHF           ;  save registers
        PUSH   SI       ;  used by
        PUSH   CX       ;  subroutine

        MOV    SI,CURAD ;  point to current line
        ADD    SI, OFFSET ;
        CLD             ;  auto increment
        MOv    CX,40
  REP   LODS   WDPTR    ;  move 40 words
        CMP    SI, LAST ;  check for end of screen
        JNZ    KTPK     ;  jump if not at end
        MOv    SI,TOPDIS ;  end-set to top
KTPK    MOv    CURAD,SI

        POP    CX       ;  restore
        POP    SI       ;
        POPF            ;
```

7. A screen-refresh routine illustrates how the iAPX 88/10 load-string (LODS) instruction fills an 8276 row buffer. The 15 lines take 167 μs and are run every ten CRT lines (every 617 μs).

```
XOR     AX, AX          ;  clear AX
MOV     BX, ESCTBL      ;  load table, pointer
MOV     AL, USCHR       ;  read character
CMP     AL, 41H         ;  check for 41H
JL      SETUP           ;  not valid
CMP     AL, 48H         ;  check for 48H
JG      SETUP           ;  not valid
XLAT                    ;  translate to routine
                           address
JMP     (AX)
```

8. This routine checks the keyboard character to see if it is a valid escape-sequence command (41H through 48H). If the character is valid, a translate table jumps to a service routine. With the powerful iAPX 88/10 translate instruction, the service routine takes just 7 μs.

hardware automatically sends a write (over buffer-select and write pins) to the 8276.

The simultaneous memory-read and 8276-write commands transfer characters from the 8185 RAM to the 8276 in a single memory cycle—without a direct-memory-access (DMA) controller. The 80 reads are under the control of the CPU load string (LODS) instruction, which handles 40 word loads with iAPX 88/10 code (Fig. 7). The complete refresh sequence for one line requires approximately 167 μs. As a result, processor overhead for refresh operations is approximately 27%.

Foreground software also involves keyboard scanning that is performed only at the end of each display frame (after 25 rows or 16.7 ms). If a key depression is noted during one of these scans, the information is stored for further background processing. An iAPX 88/10 routine checks the character to determine whether it is a valid escape-character command (Fig. 8). In this procedure, the iAPX 88/10's translate instruction (XLAT) takes care of table lookup.

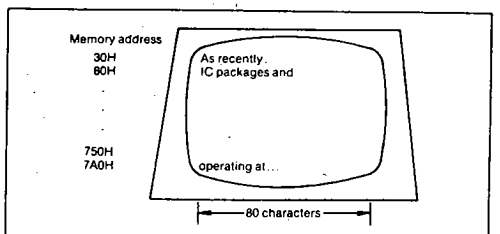
The low-priority software section handles "background" processing. It monitors the 8251A serial I/O port and provides processing for characters entered via the keyboard or with the serial interface. Background software executes continuously except when interrupted for the higher-priority foreground processing.

Cumbersome scrolling technique avoided

A refresh-buffer memory stores all 2000 characters that can be displayed on the CRT screen. The foreground software transfers one row (of 80 characters) at a time to the 8276. Two pointers are employed during normal operation. Under the control of foreground processing software, the current-row pointer contains the address of the next row to be displayed. This pointer must always be correct, so that a row can be transferred to the 8276 when requested. The buffer pointer contains the address of the next CRT buffer location to be written into (from either the keyboard or the serial port). Controlled by the background software, the buffer pointer indicates the cursor's actual location.

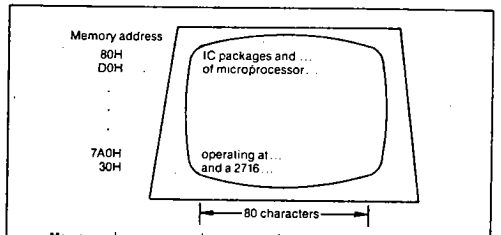
The simplest refresh-buffer organization associates the first memory address with the upper left position on the CRT screen. All other characters are stored sequentially (Fig. 9). But this method makes CRT screen scrolling difficult. Scrolling requires that each display line be moved up one row. The top line of the CRT is lost, the bottom line is blanked, and the cursor is placed at the beginning of the bottom line.

With this fixed sequential organization, all characters in the refresh buffer must be moved forward



Memory address	ASCII	Hex	Notes
30H	A	41	First character, first row
31H	s	73	
32H	space	20	
80H	I	49	First character, second row
81H	C	43	
7FFH	space	20	Last screen character

9. This memory/screen-character relationship exists when all characters are stored sequentially, making scrolling difficult.



Memory address	ASCII	Hex	Notes
30H	a	61	First character, 25th row
31H	n	6E	
32H	d	64	
80H	I	49	First character, first row
81H	C	43	
7FFH	space	20	Last character, first row

10. If sequential memory orientation is retained but characters do not have to be moved in memory, scrolling can be much more efficient. Here, scrolling is accomplished simply by changing the display-start pointer. The memory/screen-character relationship is shown after a scroll of one line from the positions illustrated in Fig. 9.

by 80 characters (memory locations) to scroll the screen. (Each line moves up one row on the CRT and the last 80 characters in the buffer are blanked.) Moving 1920 characters each time the screen scrolls a single line is very slow and cumbersome.

The low-cost CRT controller avoids this problem with a slight modification of the fixed-sequential scrolling technique. Here, sequential memory orientation is retained while the need to move characters in memory is eliminated. This approach requires an additional display-start pointer that points to the memory location of the first character to be displayed.

At system initialization, the display-start pointer is set to 30H, the buffer-start address. During each vertical-retrace interval, the current-row pointer is initialized from the display-start pointer. Scrolling is performed by merely changing the display-start pointer.

For a single row scroll, the display-start pointer moves ahead 80 characters to location 80H, and the first 80 characters in the buffer are blanked. During the next vertical retrace, the foreground software sets the current-row pointer to the display-start location (80H), and begins transferring characters to the 8276 from this address.

The character in memory-location 80H (previously the first character in the second row) now occupies the first display position on the CRT screen (first character of the first row). When the foreground software reaches the end of the display buffer, the next row is read from the beginning of the buffer (location 30H). Thus, the first 80 characters in the buffer appear on the last display row (Fig. 10).

Each subsequent scroll moves the display start pointer forward by 80 characters. Buffer operations automatically "roll over" to the physical beginning of the buffer after passing the last buffer location.

Since the row-by-row character display is controlled by iAPX 88/10 software, other display techniques may be used. In particular, a linked list structure is extremely adaptable to word-processing and text-editing functions. This method allows each row within a file to be changed independently of other rows.

Because the rows are linked or "chained together" by pointers, rows may be easily inserted or deleted by simply changing pointers. To display a CRT frame, the processor simply follows the pointer chain from one row to the next.□